

# **BI-DIRECTIONAL ASSOCIATIVE MEMORY NETWORKS**

---

*EECS 6980: Neural Networks: Theory and RF/Microwave*

**EDRIS AMIN**

**APRIL 12, 2012**

## Introduction

The Bi-directional Associative Memory (BAM) Network will be covered today. Bart Kosko first introduced the idea of BAM in [1]. His attempt was one of the more popular ones during an era when many researchers were attempting to develop a psychological model for computation [2]. The ultimate goal for psychological computation is to create a model which requires a smaller training set and provides improved accuracy despite being provided a limited training set [2].

Other associative memory networks include the Hopfield and Hebbian network models [3, 4]. Both Hopfield and Hebbian networks associate inputs with outputs of equal size. Mapping elements to elements of equal size has some benefits including higher memory capacity. A weakness of mapping sets of equal size is that as the set size grows the number of matrix calculations grow exponentially.

The BAM network differs in that it maps two sets of arrays with unequal size, known as “heteroassociative” [4]. This has a smaller maximum capacity associated with the size of the smaller arrays in one of the sets. In the world of computing, a BAM network may be desirable because it requires fewer computations than a Hopfield network [4].

Throughout this paper the number of input arrays will always be equal to the number of output arrays being memorized; only the size of arrays may vary with BAM.

i.e. consider figure 1.

If figure 1 were to represent a Hopfield network then  $\text{length}(y_j) = \text{length}(x_i)$

If figure 1 were to represent a BAM network then  $\text{length}(y_j) < \text{length}(x_i)$

## Hebb Rule for Pattern Association

The Hebb Rule is commonly used for determining the weights of Associative Memory Networks. The Hebb Rule can be applied for networks with binary sets and bipolar sets [3]. The algorithm considers associating set X with set Y. A weight,  $w_{ij}$  is associated with the weight connecting element  $x_i$  to element  $y_j$ . This relationship is shown in figure 1.

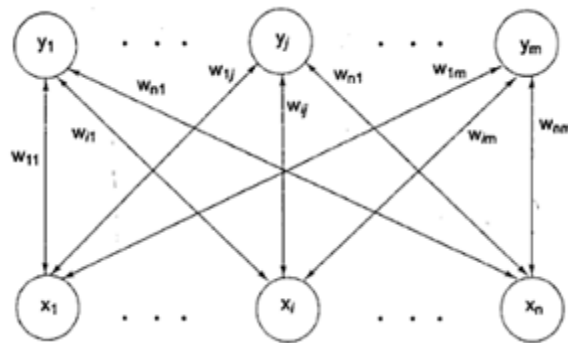


Fig. 1

Hebb Rule for calculating weights in a BAM network with P arrays to memorize [4];

$$[W] = \sum_{p=1}^P ([X_p][Y_p]^T)$$

## Binary Vs Bipolar

In [3] it was mentioned that BAM networks can have binary or bipolar forms. In a binary BAM network each of the elements  $x_i$  and  $y_j$  have binary values 0, or 1. In a bipolar BAM each of the elements  $x_i$  and  $y_j$  have bipolar values -1, or 1. How these values are determined is explained in the following section.

## BAM Application

Associate set X with set Y

$$X1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \leftrightarrow Y1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}; X2 = \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \leftrightarrow Y2 = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}; X3 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \end{bmatrix} \leftrightarrow Y3 = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}; X4 = \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \leftrightarrow Y4 = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$$

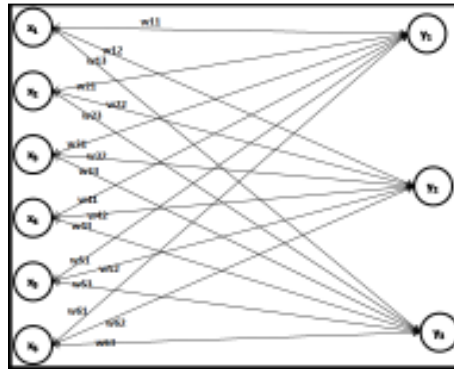


Fig2. Representation of BAM network application example

**Step 1.** Determine weight matrix W using Hebb rule.

$$W = X1 * Y1^T + X2 * Y2^T + X3 * Y3^T + X4 * Y4^T$$

$$= \begin{bmatrix} 4 & 0 & 4 \\ 4 & 0 & 4 \\ 0 & 4 & 0 \\ 0 & 4 & 0 \\ 4 & 0 & 4 \\ 4 & 0 & 4 \end{bmatrix}$$

**Step 2.** Activation function (can be applied in directions)

When the BAM is given an input from the set X it should produce the appropriate output from the set Y where  $Y = [y_1 \ y_2 \ y_3]^T$

$$y_j = x_1 w_{1j} + x_2 w_{2j} + x_3 w_{3j} + x_4 w_{4j} + x_5 w_{5j} + x_6 w_{6j}$$

$$y_j = \text{sign}(y_j) \quad \rightarrow \quad Y_p = \text{sign}(W^T * X_p)$$

When the BAM is given an input from the set Y it should produce the appropriate output from the set X where  $X = [x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6]^T$

$$x_i = y_1 w_{i1} + y_2 w_{i2} + y_3 w_{i3}$$

$$x_i = \text{sign}(x_i) \quad \rightarrow \quad X_p = \text{sign}(W * Y_p)$$

The output of the  $\text{sign}()$  function is determined by whether the system is binary or bipolar

Binary

$$\text{sign}(t) = \begin{cases} 1 & \text{if } t > 0 \\ 0 & \text{if } t \leq 0 \end{cases};$$

Bipolar

$$\text{sign}(t) = \begin{cases} 1 & \text{if } t > 0 \\ 0 & \text{if } t = 0 \\ -1 & \text{if } t < 0 \end{cases}$$

When we use binary our matrix calculations will be easier because we have more cases of multiplying by 0.

**Step 3.** Unknown inputs are calculated in the same way.

Take  $X5 = [1, -1, -1, -1, -1, -1]^T$

This produces  $Y5 = \text{sign}(\mathbf{W}^T * X5) = [-1 \ -1 \ -1]^T$

In [4] it is mentioned that when an unknown vector is not associated with a corresponding memorized vector than the new vector should be run through the BAM in the reverse direction. This should be performed until a convergence is achieved.

1. Take  $X6(0)$  producing  $Y6(0)$  after the first iteration
2. If  $Y6(0)$  is not a good match to a memorized state then calculate  $X6(1)$  using  $Y6(0)$  as input
3. If  $X6(1)$  is not a good match to a memorized state then calculate  $Y6(1)$  using  $X6(1)$  as input
4. Repeat 2 and 3 until a desirable match is found

## Comparison Hopfield vs BAM

### Hopfield method

Memorize set P

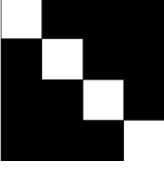
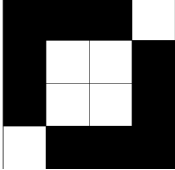
<b>P1</b> = [1 1 1 1 1 -1 -1 1 1 -1 -1 1 1 1 1 1]; 	<b>P2</b> = [1 -1 -1 1 -1 1 1 -1 -1 -1 1 1 -1 -1 -1 1]; 
--	---

$$\mathbf{W} = \mathbf{P1} * \mathbf{P1}^T + \mathbf{P2} * \mathbf{P2}^T - 2 * \mathbf{I}_{16} =$$

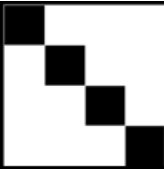
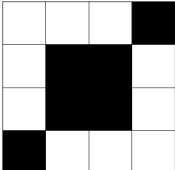
0	0	0	2	0	0	0	0	0	0	0	0	2	0	0	2
0	0	2	0	2	-2	-2	2	2	-2	-2	2	0	2	2	0
0	2	0	0	2	-2	-2	2	2	-2	-2	2	0	2	2	0
2	0	0	0	0	0	0	0	0	0	0	0	2	0	0	2
0	2	2	0	0	-2	-2	2	2	-2	-2	2	0	2	2	0
0	-2	-2	0	-2	0	2	-2	-2	2	2	-2	0	-2	-2	0
0	-2	-2	0	-2	2	0	-2	-2	2	2	-2	0	-2	-2	0
0	2	2	0	2	-2	-2	0	2	-2	-2	2	0	2	2	0
0	2	2	0	2	-2	-2	2	0	-2	-2	2	0	2	2	0
0	-2	-2	0	-2	2	2	-2	-2	2	2	-2	0	-2	-2	0
0	2	2	0	2	-2	-2	2	2	-2	-2	2	0	2	2	0
2	0	0	2	0	0	0	0	0	0	0	0	2	0	0	2
0	2	2	0	2	-2	-2	2	2	-2	-2	2	0	2	2	0
0	2	2	0	2	-2	-2	2	2	-2	-2	2	0	2	2	0
2	0	0	2	0	0	0	0	0	0	0	0	2	0	0	2

Hopfield Weight Calculation Computations:  $2*(16*16) + 16 = 528$  multiplications

Test with unknown input  $A_{test}$

$A_{test} = [1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1];$ 	$Output = \text{sign}(W^T * A_{test}) = [-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]$  Associates with <b>P2</b>
---	--

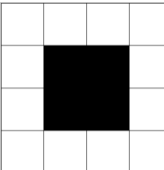
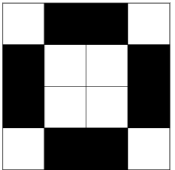
Test with unknown input  $-A_{test}$

$-A_{test} = [-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1];$ 	$Output = \text{sign}(W^T * -A_{test}) = [1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]$  Associates with <b>P1</b>
---	--

Hopfield Test Computation:  $16^2 = 256$  multiplications

### BAM method

Associate  $P \leftrightarrow B$

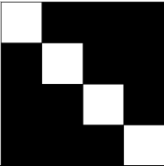
$P1 = [1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1];$ 	$P2 = [1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1];$ 
$P1 \leftrightarrow B1 = [-1 -1 -1 -1]$	$P2 \leftrightarrow B2 = [1 -1 -1 -1]$

$$W = P1 * B1^T + P2 * B2^T =$$

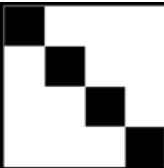
0	-2	-2	0	-2	2	2	-2	-2	2	2	-2	0	-2	-2	0
0	-2	-2	0	-2	2	2	-2	-2	2	2	-2	0	-2	-2	0
0	-2	-2	0	-2	2	2	-2	-2	2	2	-2	0	-2	-2	0

BAM Weight Calculation Computation:  $2*(3*16) = 96$  multiplications

Test with unknown input  $\mathbf{A}_{\text{test}}$

$\mathbf{A}_{\text{test}} = [1 -1 -1 -1 -1 1 1 -1 -1 -1 -1 -1 -1 -1 1];$ 	<p>Output = <math>\text{sign}(\mathbf{W}^T * \mathbf{A}_{\text{test}}) = [1 \ 1 \ 1] = \text{B2}</math></p> <p>Match <math>\mathbf{A}_{\text{test}}</math> to Pattern P2</p>
---	--

Test with unknown input  $-\mathbf{A}_{\text{test}}$

$-\mathbf{A}_{\text{test}} = [-1 \ 1 \ 1 \ 1 \ 1 -1 \ 1 \ 1 \ 1 \ 1 -1 \ 1 \ 1 \ 1 \ 1 -1];$ 	<p>Output = <math>\text{sign}(\mathbf{W}^T * \mathbf{A}_{\text{test}}) = [-1 \ -1 \ -1] = \text{B1}</math></p> <p>Match <math>-\mathbf{A}_{\text{test}}</math> to Pattern P1</p>
---	--

BAM Test Computation:  $16 * 3 = 48$  multiplications

## References

- [1] B. Kosko, "Bidirectional associative memories" *IEEE Transactions on Systems, Man, and Cybernetics, SMC-17* 1987, pp. 49-60
- [2] J.A. Anderson, "Cognitive and psychological computation with neural models" *IEEE Transactions on Systems, Man, & Cybernetics*, Vol 13(5), Sep-Oct 1983, 799-815.
- [3] S.N. Sivanandam, S. Sumathi, S.N. Deepa, *Introduction to Neural Networks Using Matlab 6.0*, Tata McGraw-Hill Education, 2006
- [4] M. Negnevitsky, *Artificial Intelligence A Guide to Intelligent Systems*, 2<sup>nd</sup> Edition, Addison Wesley, 2005
- [5] R. Krulwich (2012, March 30) "Neuroscientists Battle Furiously Over Jennifer Aniston" *NPR*  
Available: <http://www.npr.org/blogs/krulwich/2012/03/30/149685880/neuroscientists-battle-furiously-over-jennifer-aniston>
- [6] A. Abbott (2009) "Neuroscience: Opening up brain surgery." **Nature** 461 pp. 866–888.  
Available: <http://www.nature.com/news/2009/141009/full/461866a.html>